Clustering algorithms on GPU

Team 18

DEEPAM SARMAH

1 INTRODUCTION

In the day and age of data science, there has been an increased demand for clustering algorithms. Clustering algorithms cluster similar data points based on their characteristics or features. They are helpful in many cases. To name a few, they help segment the data into meaningful groups based on a standard metric (these can be maximum revenue, average life expectancy, etc.) and anomaly detection, where clustering algorithms can help identify outliers or anomalies in the data by identifying data points that do not belong to any cluster (this is helpful in cancer detection). GPUs help increase the efficiency of clustering algorithms mainly due to the following: Many clustering algorithms are computationally intensive, and hence the GPU could parallelize these computations. Another benefit is that GPUs provide a larger memory bandwidth and processing power, making GPUs ideal for clustering large data-sets. In this project, I aim to compare the serial and parallel implementations of K-Means, DBSCAN, and Mean-Shift clustering algorithms and compute speedup statistics on the same.

2 LITERATURE REVIEW

There are several clustering algorithms such as K-Means, DBSCAN and Mean-Shift algorithm. I reviewed their research papers and implementations online.

2.1 K-Means Algorithm

The K-Means algorithm was first proposed by MacQueen in 1967 [1] and then later enhanced by Hartigan and Wong [2] in 1979. The K-Means algorithm is an unsupervised clustering algorithm in which the algorithm starts by randomly selecting a number of data points as the initial centroids for each cluster. Then, each data point in the dataset is assigned to the cluster whose centroid is closest to it. Next, the centroids are recalculated as the mean of all data points in the cluster. This process of assigning points to the closest cluster and recalculating centroids is repeated until a maximum number of iterations is reached. The result is k clusters, each represented by a centroid, where the data points in each cluster are similar to each other and different from those in the other clusters. There have been implementations of K-Means on GPUs using CUDA by Reza Farivar, et al. [3] in 2008. The K-Means algorithms is useful as it can handle large data-sets and can converge to the solution quickly.

2.2 DBSCAN Algorithm

Density-based spatial clustering of applications with noise (DBSCAN) is a data clustering algorithm proposed by Martin Ester, et al. [4] in 1996. A GPU implmentation of DBSCAN was proposed by Guilherme Andrade, et al. [5] in 2013. DBSCAN works by defining two important parameters: the radius ϵ and the minimum number of points required to form a cluster (say Φ). It starts with a random data point, and builds a cluster around it by identifying all points within a distance of ϵ. If the number of points in this region is greater than or equal to Φ, then a new cluster is formed. Otherwise, the point is marked as noise or an outlier. DBSCAN then continues to grow the cluster by recursively finding new points within the ϵ radius and adding them to the cluster until no new points can be added. The algorithm repeats this process for all unvisited points in the dataset, assigning them to clusters or marking them as noise. DBSCAN is highly useful as it can handle non-spherical clusters and can identify the number of clusters based on the density of the data.

Author’s address: Deepam Sarmah, deepam20050@iiitd.ac.in.

2.3 Mean-Shift Algorithm

A description of the Mean-Shift algorithm is found in the paper by K Fukunaga, et al. [6] in 1975. The mean shift algorithm works by by randomly selecting a data point and defining a window around it. The mean of all data points within the window is then computed, and the window is shifted towards this mean. This process is repeated for each window until the window no longer moves or changes significantly. At the end of the process, each data point is assigned to the nearest converged window, which represents a cluster. The Mean-Shift algorithm is advantageous as it can handle large data-sets with many features and is suitable for clustering data-sets with complex cluster shapes.

3 MILESTONES

The identified milestones are:

S. No.	**Milestone**	**Member**
1	Implement K-Means on CPU	Deepam
2	Implement Mean-Shift on CPU	Deepam
3	Implement K-Means and Mean-Shift on GPU	Deepam
4	Implement DBSCAN on CPU	Deepam
5	Implement DBSCAN on GPU	Deepam
6	Compare and analyze GPU code using profiler	Deepam

REFERENCES

[1] J MacQueen. Classification and analysis of multivariate observations. In *5th Berkeley Symp. Math. Statist. Probability*, pages 281–297. University of California Los Angeles LA USA, 1967.

[2] John A Hartigan, Manchek A Wong, et al. A k-means clustering algorithm. *Applied statistics*, 28(1):100–108, 1979.

[3] Reza Farivar, Daniel Rebolledo, Ellick Chan, and Roy H Campbell. A parallel implementation of k-means clustering on gpus. In *Pdpta*, volume 13, pages 212–312, 2008.

[4] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD'96, page 226–231. AAAI Press, 1996.

[5] Guilherme Andrade, Gabriel Ramos, Daniel Madeira, Rafael Sachetto, Renato Ferreira, and Leonardo Rocha. G-dbscan: A gpu accelerated algorithm for density-based clustering. *Procedia Computer Science*, 18:369–378, 2013.

[6] Keinosuke Fukunaga and Larry Hostetler. The estimation of the gradient of a density function, with applications in pattern recognition. *IEEE Transactions on information theory*, 21(1):32–40, 1975.